Ruby Classes

You will learn how classes are implemented in the Ruby language and how they are different.

Ruby Classes

- ~~Open classes~~
- ~~Duck Typed~~
- ~~Just a runnable code~~
- Object themselves*
- Modules with hierarchies
- Scopes
- Object model
- Etc.

Insight 5:

Classes are objects

Classes in ruby are actually objects. In other words, instances of Class class.

Does any of these sound familiar?

- Eigenclass
- Singleton class (not to be confused with Singleton design pattern)
- Classes are objects/instances themselves

Classes are objects

Classes are an instances of a class named Class.

Classes are objects

```ruby
class MyClass
end
```

```ruby
a = MyClass.new
puts a.class                    # MyClass

MyClass.class                   # Class
Class.class                     # Class
```

Classes are objects

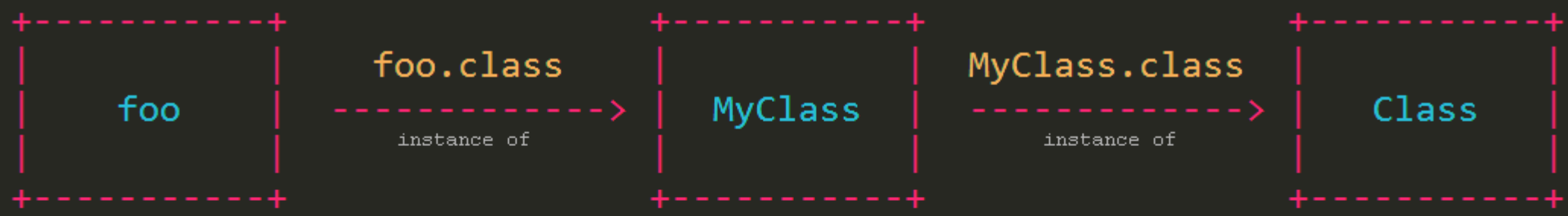
```ruby
MyClass = Class.new

MyClass.name                    # MyClass
MyClass.class                   # Class

a = MyClass.new
a.class                         # MyClass


foo = Class.new

foo.name                        # nil
foo.class                       # Class

Bar = foo
foo.name                        # Bar
```

Classes are objects

```ruby
class MyClass; end
foo = MyClass.new


foo     = MyClass.new       # "foo" is an instance of MyClass class.

MyClass = Class.new         # "MyClass" is a constant that holds an
                            # instance of "Class" class.

foo = (Class.new).new       # So, "foo" is an instance of an instance
                            # of a "Class" class
```

Visualizing it

```ruby
class MyClass; end
foo = MyClass.new
```



"foo" is an Object
Also, an instance
of MyClass

"MyClass" is a class
Also, an instance
of class

"Class" is a class
Also, an instance
of itself

Quick recap:

- Classes in ruby are instances of class "`Class`"
- Class names are constants
- Everything that applies to instances of classes, also applies to classes (instance variables, methods, etc.)

Class' class is called:

- Eigenclass
- Singleton class

How can we access class' class (eigenclass)?

```ruby
class MyClass

    class << self
        def class_method
            # Eigenclass method
        end
    end

    def self.class_method2
        # Alternative method to
        # define methods in eigenclass
    end
end

def MyClass.class_method3
    # Another alternative
end
```

How can we access eigenclass? (cont.)

```ruby
class MyClass
    class << self                           # entering eigenclass
        def class_method                    # class method (MyClass.class_method)
            @foo = 1
        end
    end

    def self.foo                            # class method (MyClass.foo)
        @foo
    end

    def foo                                 # instance method (MyClass.new.foo)
        @foo
    end
end

def MyClass.class_method3
    @foo = 2
    @bar = "Another class instance var"
end


foo = MyClass.new

MyClass.class_method                        # Sets class instance var: @foo = 1

MyClass.foo                                 # Returns: 1
foo.foo                                     # Returns: nil

MyClass.class_method3                       # Sets class instance var: @foo = 2 and @bar

MyClass.foo                                 # Returns: 2
foo.foo                                     # Returns: nil

foo.class_method                            # NoMethodError: undefined method `class_method' for #<MyClas
```

Instance & class instance vars access

```ruby
class MyClass
    def initialize
        @foo = "Instance variable"
    end

    def instance_method
        @foo
        # can access instance vars: @foo
        # can't access @bar
    end

    class << self
        def singleton_method
            # can't access @foo
            # can define own instance vars
            @bar = "Class instance variable"
        end
    end
end

my_class = MyClass.new

my_class.instance_method                # "instance variable"
my_class.singleton_method               # NoMethodError: undefined method `singleton_method' for #<MyClass:0x1fc6de8>

MyClass.instance_method                 # NoMethodError: undefined method `instance_method' for MyClass:Class
MyClass.singleton_method                # "Class instance variable"
```

Recap:

- Classes in ruby are instances of class "`Class`"
- Class names are constants
- Everything that applies to instances of classes, also applies to classes (instance variables, methods, etc.)
- Class' class is called eigenclass or singleton class
- Eigenclass methods are similar to static methods
- Instance methods and eigenclass/static methods live in different scopes and don't share variables

Our progress so far:

- ~~Open classes~~
- ~~Duck Typed~~
- ~~Just a runnable code~~
- ~~Object themselves~~*
- Scopes
- Modules with hierarchies
- Inheritance hierarchy
- Ruby object model
- Method lookup
- Etc.